OpenJUMP ja OpenStreetMap-reititys

Jukka Rahkonen, http://latuviitta.org

Viimeksi muokattu 12. huhtikuuta 2013

Tiivistelmä

Verkossa toimivat reitityspalvelut ottavat vastaan reitin alku- ja loppupisteen ja palauttavat sitten reitin. Lisähienouksina voi olla reititystavan valinta, esimerkiksi nopein tai lyhin reitti, tai autoiluun, pyöräilyyn tai jalankulkuun parhaiten sopiva reitti.

Reitityspalvelun käyttöön tarvitaan siis kolme rakennuspalikkaa:

1. Väline, jolla voidaan antaa reitin alku- ja loppupisteet.
2. Väline, joka pyytää reitin reitityspalvelusta.
3. Väline, joka näyttää reitin.

OpenJUMP-karttaohjelman avulla voidaan tehdä helposti mallisuoritus reitityspalveluiden käytöstä.

Valmistelut

Haetaan ja asennetaan OpenJUMP-ohjelmisto osoitteesta http://openjump.org

Tallennetaan tämän ohjeen liitteessä 1. oleva BeanShell koodi tekstitiedostoksi, jolle annetaan nimi ”YOUR_routins.bsh”. Viedään tämä tiedosto OpenJUMP:in Bean-työkaluhakemistoon ”\lib\ext\BeanTools”. Samalla tavalla tallennetaan liitteen 2. koodi nimellä Get_HSL_Route.bsh.

Vaihe 1: Alku- ja loppupisteiden antaminen

Haetaan ensiksi hieman vektoritietoja, niin pystymme sijoittamaan itsemme kartalle alku- ja loppupisteiden antamista varten. Tässä esimerkissä koordinaatistona on käytettävä maantieteellisiä leveys- ja pituusasteita (EPSG:4326), mistä syystä vektoreiksi sopivat erinomaisesti OpenStreetMap-projektin shapefile-muotoiset vektorit, jotka voidaan ladata osoitteesta

http://download.geofabrik.de/europe/finland-latest.shp.zip

Geofabrikin zip-pakettiin kuuluu monia eri aineistoja. Tässä harjoituksessa emme tarvitse niitä kaikkia, ja siksi voidaan myös käyttää GDAL-ohjelmiston mahdollistamaa kätevää tapaa lukea zip-tiedoston sisältä vain haluttu shapefile. Se onnistuu seuraavalla komennolla:

```
ogr2ogr -f "ESRI Shapefle" roads.shp
/vsizip/vsicurl/http://download.geofabrik.de/europe/finland-latest.shp.zip roads
```

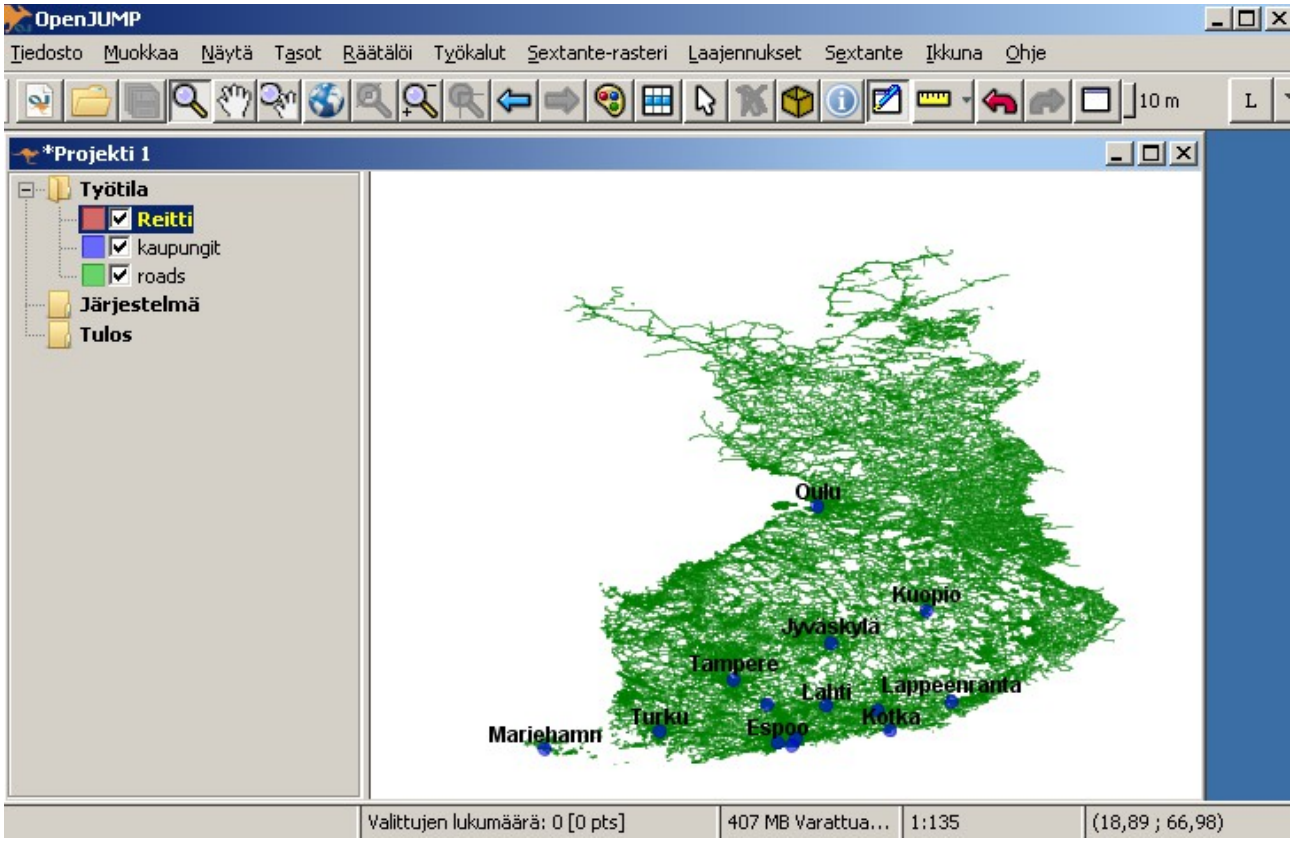
Komento hakee Geofabrikin palvelimella olevan zip-tiedoston sisältä roads-nimisen paikkatietoaineiston ja tallentaa sen paikalliselle levylle nimellä ”roads.shp”. Aineiston nimi on tunnettava; jos se ei ole ennestään tiedossa, niin listan saatavilla olevista tasoista saa komennolla

```
ogrinfo -ro /vsizip/vsicurl/http://download.geofabrik.de/europe/finland-latest.shp.zip
```

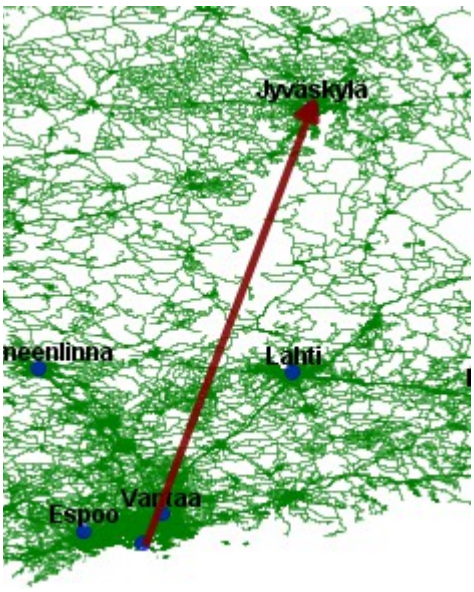
Tämän komennon käyttäminen on enemmänkin kuin hauska temppu. Se nimittäin pystyy hakemaan OSM-tiet 187 megatavun zip-arkistosta niin, että verkkoliikennettä syntyy vain 77 megatavua (tiedostokoot 4. huhtikuuta 2013), eli tämä paljon fiksumpi tapa kuin hakea koko arkisto levylle ja purkaa sitten haluttu tiedosto siitä ulos.

GDAL-työkalujen käytöstä ei tässä sen enempää. Tarkempia ohjeita on eräissä muissa Latuviitan paikkatieto-ohjeissa sekä tietysti GDAL:n omilla sivuilla http://gdal.org.

Kun OpenStreetMap-vektoreita tai jotan muuta EPSG:4324-järjestelmässä olevaa aineistoa on hankittu, niin avataan aineisto OpenJUMP:issa.

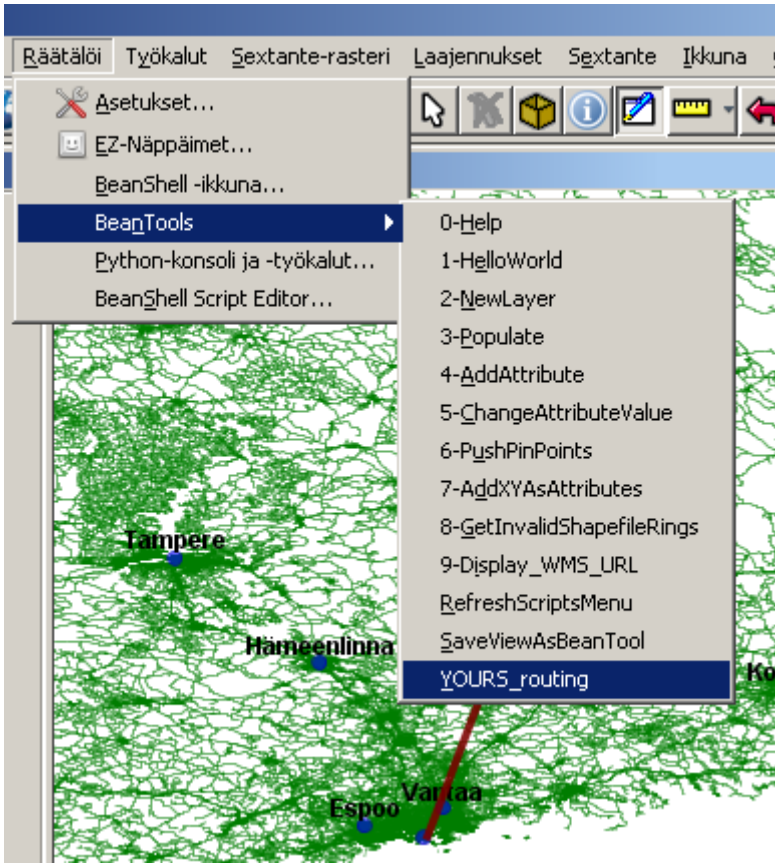


Lisätään OpenJUMP-kartalle uusi taso valikosta Tiedosto-Uusi-Taso tai valikosta, joka avautuu klikkaamalla hiiren oikealla näppeimellä karttatasoluettelon päällä. Valitaan muokkaustyökaluista työkalu ”Piirrä viiva” ja piirretään viiva kahdella pisteellä. Mutkikkaan viivan piirtämisestä useilla taitepisteillä ei ole haittaa, mutta ei hyötyäkään, koska vain viivan ensimmäistä ja viimeistä pistettä tullaan käyttämään.



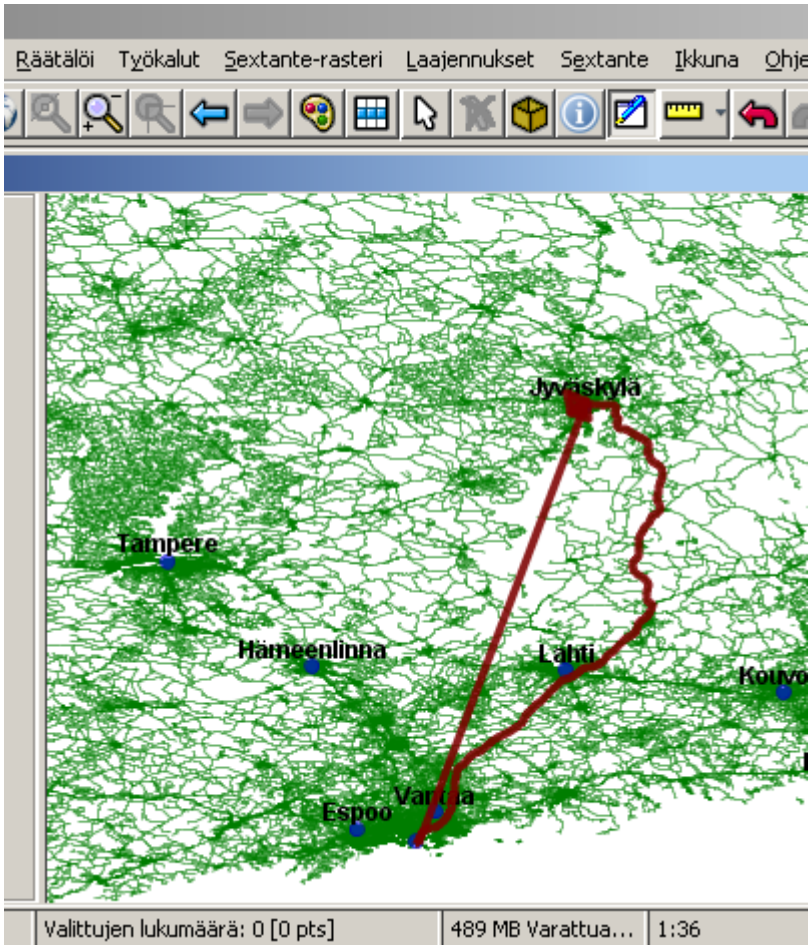
Vaihe 2. Reitin pyytäminen reitityspalvelusta

Valmisteluvaiheessa tallennettu BeanShell-skripti löytyy valikosta Räätälöi-BeanTools-YOURS_routing ja se lähettää reittipyynnön reitityspalveluun.



Vaihe 3. Reitin näyttäminen kartalla

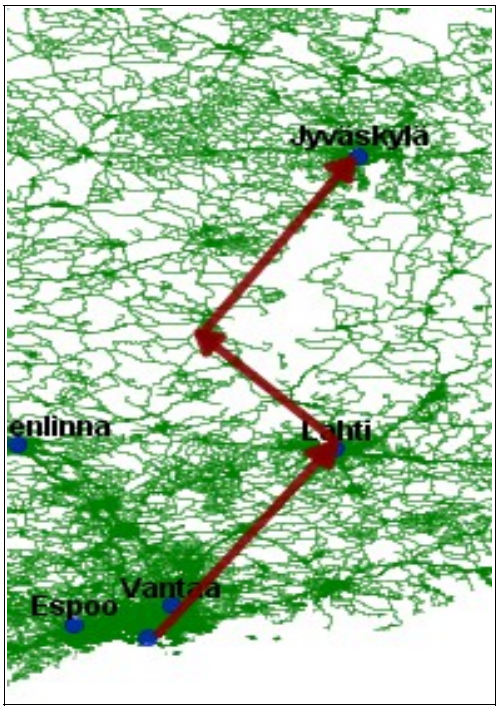
Sama BeanShell-skripti, jolla reittipyyntö lähetettiin, osaa myös piirtää palvelun lähettämän reittiehdotuksen kartalle.

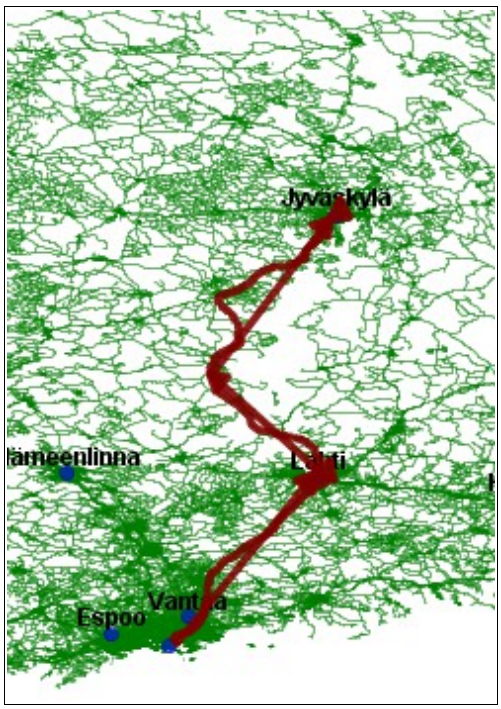


Reitin haku välietappien kautta

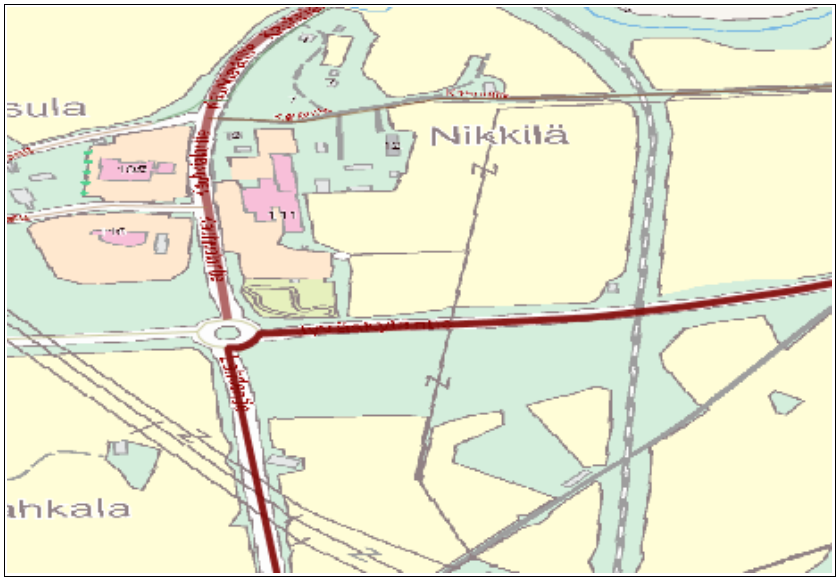
Jos halutaan etsiä reitti Helsingistä Lahden ja Kuhmoisten kautta Jyväskylään, niin se onnistuu piirtämällä kolme viivaa ja pyytämällä kolme reittiä: Helsinki-Lahti, Lahti-Kuhmoinen ja Kuhmoinen-Jyväskylä. OpenJUMP:in skripti tekee automaattisesti haun jokaisen aktiiviselta karttatasolta löytämänsä viivan alku- ja loppupisteiden perusteella. Muista siis tyhjentää taso, tai luoda uusia reittihakuja varten uusi karttataso, kun teet peräkkäisiä hakuja.

Kun haluttu reitti on piirretty, valitaan YOURS_routing OpenJUMP:in valikosta ja katsotaan, miltä tulos näyttää.

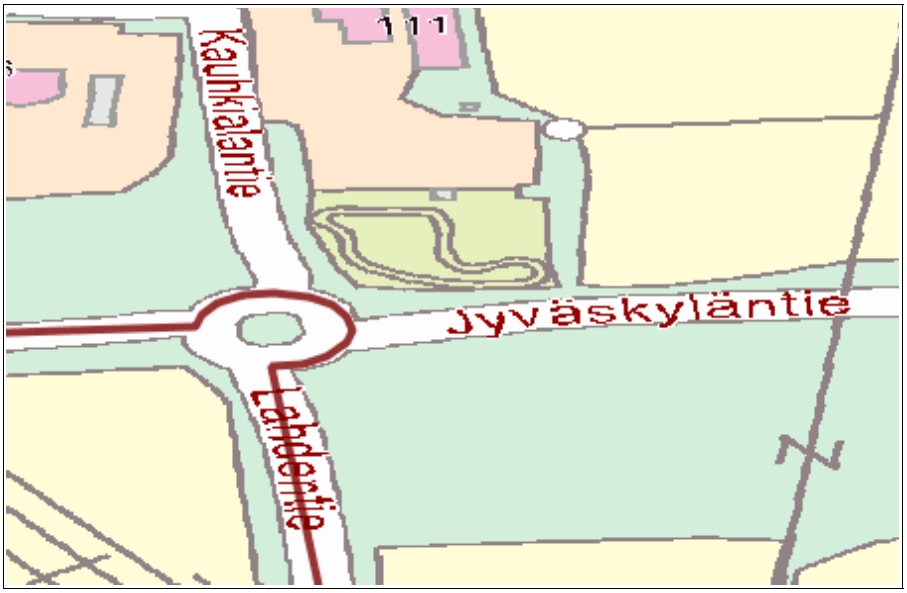




Reitti haetaan reitityspalvelusta, joka tässä tapauksessa on http://www.yournavigation.org. Reittiä ei siis haeta OpenJUMP:ssa auki olevan vektoriaineiston perusteella. Kuitenkin, koska myös yournavigation.org käyttää reititykseen samaa OpenStreetMap-aineistoa, niin reitti seurailee tarkasti OpenJUMP:ssa näkyviä vektoreita. Seuraavasta kuva antaa esimerkin tämän vektoriaineiston tarkkuudesta. Paikka on teiden 24 ja 9 risteys Jämsässä, ja reitistä huomataan, että liikenneympyrä on mallinnettu aineistoon todellakin ympyränä eikä pelkkänä risteyspisteenä.



OpenStreetMap ja yournavigation tietävät myös, että lyhin reitti ei ole aina mahdollinen, kuten tässä, kun halutaan kääntyä Lahdentieltä länteen.



Miten se toimii?

YOURS_routing.bsh -skripti luo seuraavanlaisen http-pyynnön

```
http://www.yournavigation.org/api/1.0/gosmore.php?
format=kml&
flat=61.84361791171761&
flon=25.176709841366492&
tlat=61.85041397176689&
tlon=25.16789466202721&
v=motorcar&
fast=1&
layer=mapnik
```

Kaikki muut parametrit tulevat skriptistä aina samoina, mutta flat, flon, tlat ja tlon otetaan OpenJUMP:ssa piirrettyjen viivojen alku- ja loppupisteistä. Paremetrien merkitys selviää YOURS-reititysohjelman wiki-sivulta http://wiki.openstreetmap.org/wiki/YOURS

Esimerkissä haetaan autolla ajattavaa reittiä, pyöräilyreitin voisi etsiä muuttamalla parametrin ”v” arvoksi ”bicycle”.

Ne on kaikki samanlaisia, eli esimerkkejä muista reitityspalveluista

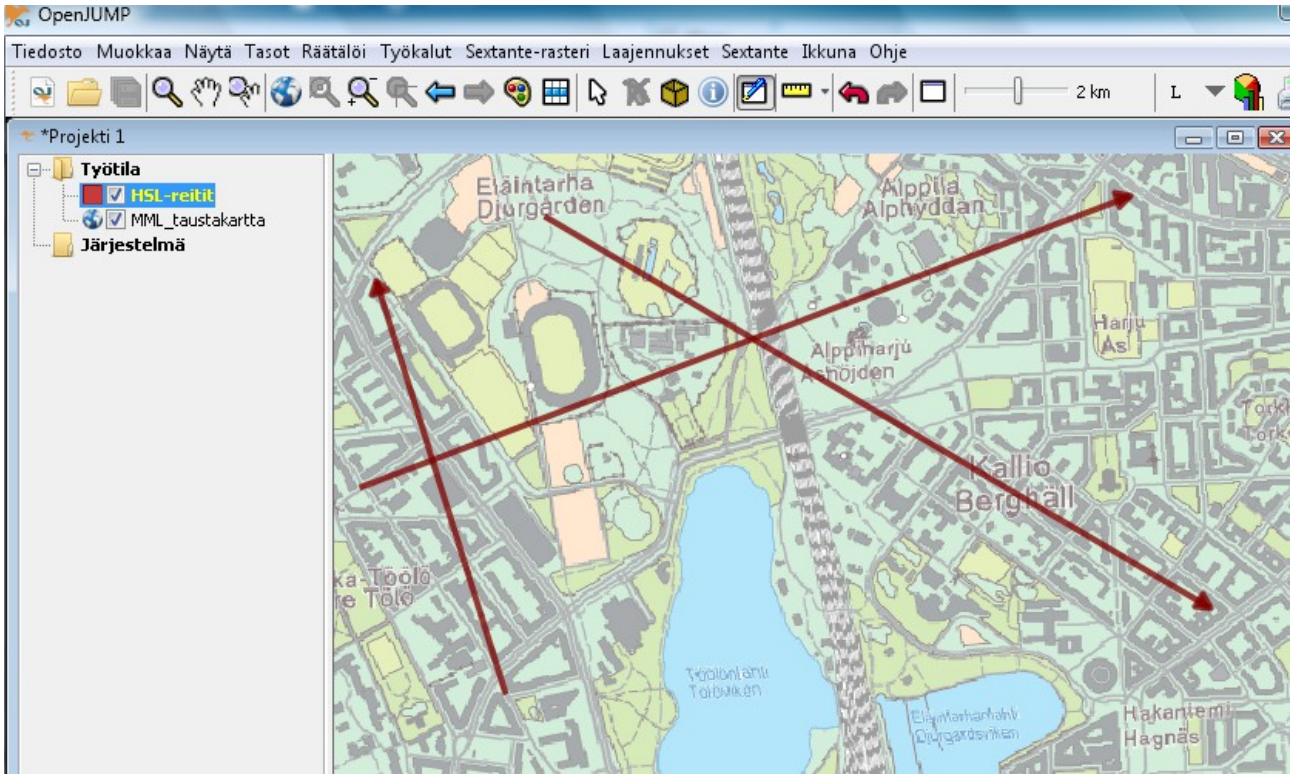
Kaikille reitityspalvelut toimivat samalla tavalla: niille annetaan alku- ja loppupisteet koordinaatteina (joko suoraan, tai sitten antamalla osoite, joka muunnetaan koordinaateiksi) ja mahdollisesti jotain lisäparametrejä. Otetaan toiseksi esimerkiksi pyöräilyn ja kävelyn reittiopas,

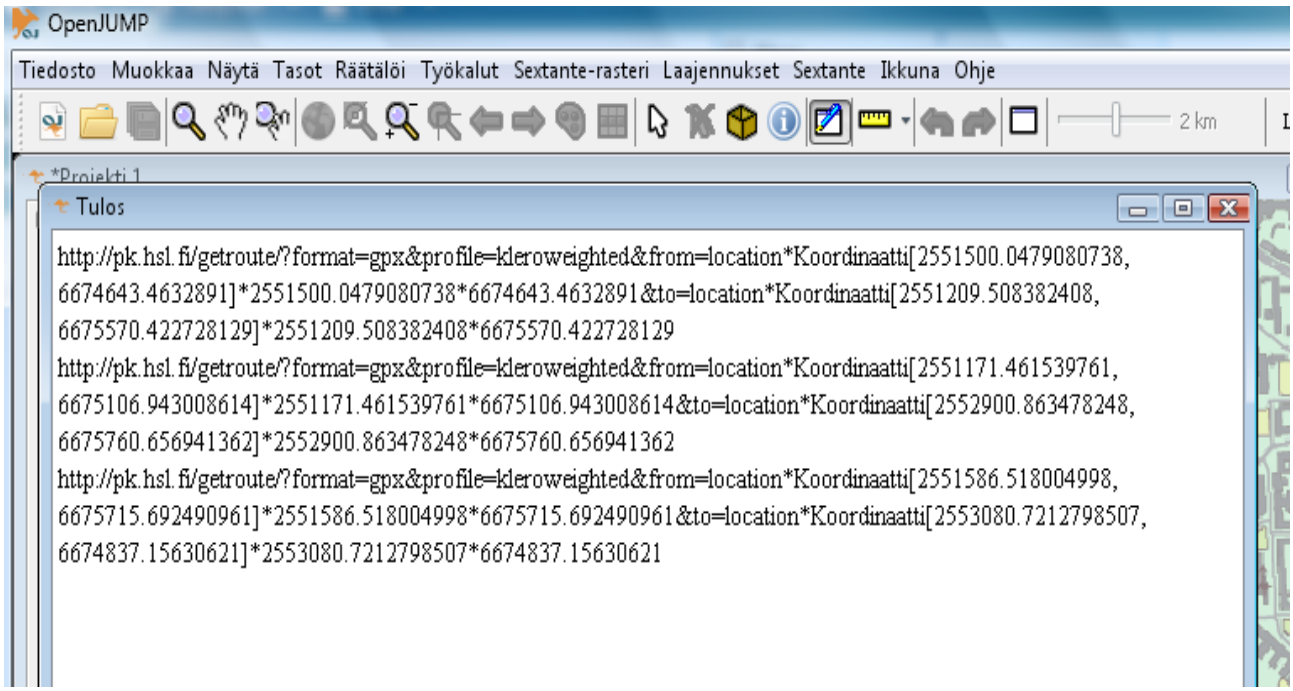
joka löytyy osoitteesta http://pk.hsl.fi/. Tätä palvelua varten täytyy kehittää seuraavan kaltaisia pyyntöjä:

```
http://pk.hsl.fi/getroute/?
format=gpx&
profile=kleroweighted&
from=location*Koordinaatti[2552242,6674964]*2552242*6674964&
to=location*Koordinaatti[2554424,6681402]*2554424*6681402
```

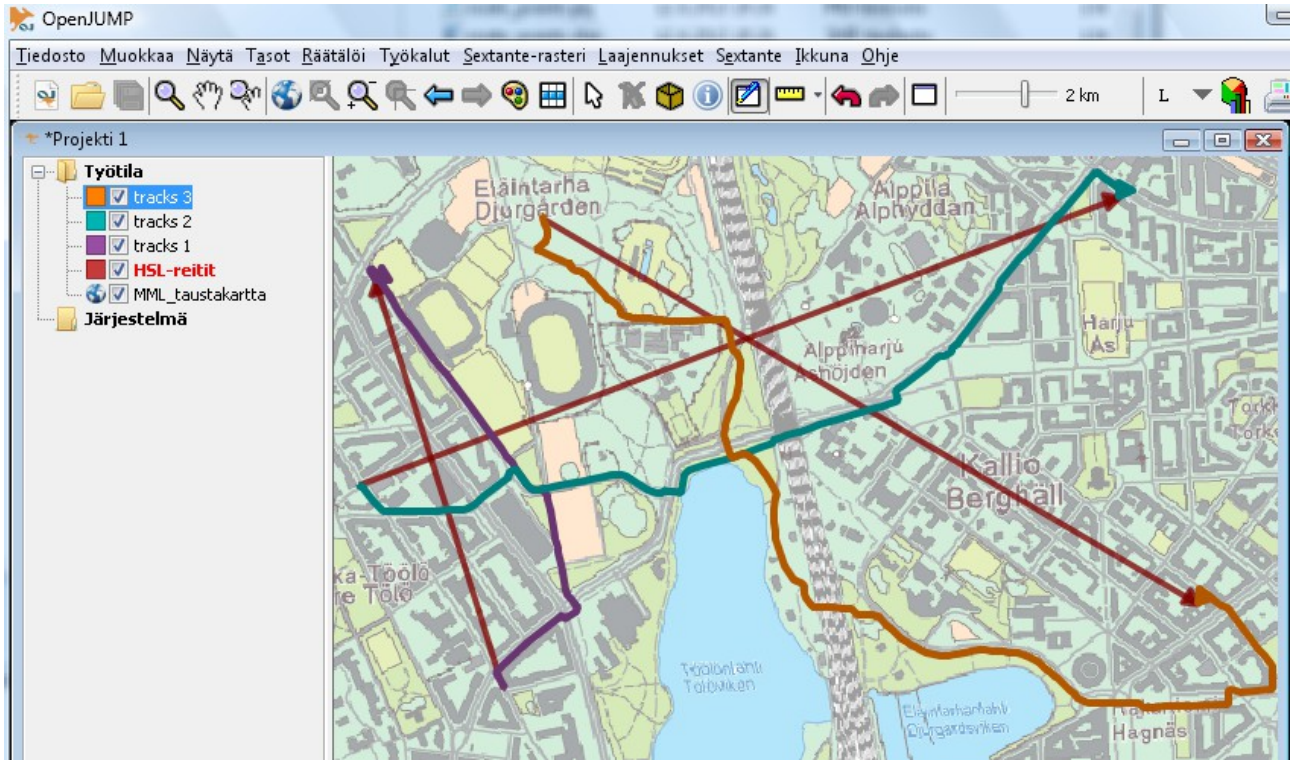
Tästä pyynnöstä täytyy huomata vain se, että koordinaatit näyttävät olevan Helsingin järjestelmän koordinaatteja, eli KKJ:n 2-kaistaa (EPSG:2392.

Tallennetaan OpenJUMP:in BeanTools hakemistoon tällä kertaa liittessä 2.oleva skripti ”Get_HSL_Route.bsh” ja hankitaan OpenJUMP:iin KKJ:n 2-kaistassa olevaa taustaaineistoa.





Tulos ei tällä kertaa näy kartalla, koska OpenJUMP ei osaa avata GPX-tiedostomuodossa tulevia reittiohjeita. Sen sijaan skripti on tehty kirjoittamaan tulosruutuun pyynnöt, jotka voidaan lähettää reitityspalveluun vaikkapa selaimella tai wget- tai curl-ohjelmalla.



Tässä kuvassa näkyvät HSL:n palvelun lähettämät reitit, jotka on ensin tallennettu GPS-muodossa levylle ja muunnettu sitten ogr2ogr-ohjelmalla shapefile-muodoon, jotta se voidaan avata OpenJUMP:ssa.

Liite 1. YOURS_routing.bsh

```
{
import com.vividsolutions.jts.geom.*;
import com.vividsolutions.jts.io.WKTReader;
import com.vividsolutions.jump.feature.*;
import com.vividsolutions.jump.geom.EnvelopeUtil;
import com.vividsolutions.jump.workbench.model.*;
import java.io.*;
import java.net.URL;

     fc =
wc.getLayerNamePanel().getSelectedLayers()[0].getFeatureCollectionWrapper();
     paths = new ArrayList();;
     count = 0;
     for (Iterator i = fc.getFeatures().iterator(); i.hasNext();) {
         Feature feature = (Feature) i.next();
         LineString lineString = (LineString)feature.getGeometry();
         URL url;
         try {
             // get URL content
             url = new
URL("http://www.yournavigation.org/api/1.0/gosmore.php?format=kml&flat="
                 +lineString.getStartPoint().getY()+"&flon="
                 +lineString.getStartPoint().getX()+"&tlat="
                 +lineString.getEndPoint().getY()+"&tlon="
                 +lineString.getEndPoint().getX()
                 +"&v=motorcar"
                 +"&fast=1"
                 +"&layer=mapnik");
             print(url);
             URLConnection conn = url.openConnection();

             // open the stream and put it into BufferedReader
             BufferedReader br = new BufferedReader(
                                 new
InputStreamReader(conn.getInputStream()));
             String inputLine;
             StringBuffer sb = new StringBuffer();
             while ((inputLine = br.readLine()) != null) {
                 sb.append(inputLine).append("\n");
             }
             wkt = sb.toString()
                     .replaceAll("(?s).*<coordinates>\\s*", "LINESTRING(")
                     .replaceAll("(?s)\\s*</coordinates>.*", ")")
                     .replaceAll("(?s)\\s+", "@")
                     .replaceAll("(?s),", " ")
                     .replaceAll("(?s)@", ",");
             print(wkt);
             bf = new BasicFeature(fc.getFeatureSchema());
             bf.setGeometry(new WKTReader().read(wkt));
             paths.add(bf);

             br.close();
             if (count++ > 9) break;

         } catch (MalformedURLException e) {
             e.printStackTrace();
         } catch (IOException e) {
             e.printStackTrace();
         }
     }
     fc.addAll(paths);
}
```

Liite 2. Get_HSL_Route.bsh

```
{
import com.vividsolutions.jts.geom.*;
import com.vividsolutions.jump.feature.*;
import com.vividsolutions.jump.geom.EnvelopeUtil;
import com.vividsolutions.jump.workbench.model.*;

        htmlFrame = wc.workbench.frame.outputFrame;
        htmlFrame.createNewDocument();
        htmlFrame.setTitle("Output for gdal_translate");
        fc = wc.getLayerNamePanel().getSelectedLayers()[0].getFeatureCollectionWrapper();
        int j=1;
        for (Iterator i = fc.getFeatures().iterator(); i.hasNext();) {
                Feature feature = (Feature) i.next();
                LineString lineString = (LineString)feature.getGeometry();
                htmlFrame.addText("http://pk.hsl.fi/getroute/?
format=gpx&profile=kleroweighted&from=location*Koordinaatti["
                        +lineString.getStartPoint().getX()+", "
                        +lineString.getStartPoint().getY()+"]"
                        +"*"+lineString.getStartPoint().getX()
                        +"*"+lineString.getStartPoint().getY()
                        +"&to=location*Koordinaatti["
                        +lineString.getEndPoint().getX()+", "
                        +lineString.getEndPoint().getY()+"]"
                        +"*"+lineString.getEndPoint().getX()
                        +"*"+lineString.getEndPoint().getY()
                        );
                j++;
        }
        wc.workbench.frame.flash(htmlFrame);
        htmlFrame.surface();
}
```